DOI: 10.1111/cgf.13452

Applying Visual Analytics to Physically Based Rendering

G. Simons[1], S. Herholz[3], V. Petitjean[2], T. Rapp[4], M. Ament[4], H. Lensch[3], C. Dachsbacher[4], M. Eisemann[5] and E. Eisemann[2]

[1]Captain AI, Netherlands
sjrasimons@gmail.com
[2]TU Delft, Netherlands
{V.J.P.Petitjean, e.eisemann}@tudelft.nl
[3]University Tübingen, Germany
sebastianherholz@student.uni-tuebingen.de, hendrik.lensch@uni-tuebingen.de
[4]Karlsruhe Insitute of Technology, Germany
{tobias.rapp, marco.ament, dachsbacher}@kit.edu
[5]TH Köln, Germany
martin.eisemann@th-koeln.de

Abstract
Physically based rendering is a well-understood technique to produce realistic-looking images. However, different algorithms exist for efficiency reasons, which work well in certain cases but fail or produce rendering artefacts in others. Few tools allow a user to gain insight into the algorithmic processes. In this work, we present such a tool, which combines techniques from information visualization and visual analytics with physically based rendering. It consists of an interactive parallel coordinates plot, with a built-in sampling-based data reduction technique to visualize the attributes associated with each light sample. Two-dimensional (2D) and three-dimensional (3D) heat maps depict any desired property of the rendering process. An interactively rendered 3D view of the scene displays animated light paths based on the user's selection to gain further insight into the rendering process. The provided interactivity enables the user to guide the rendering process for more efficiency. To show its usefulness, we present several applications based on our tool. This includes differential light transport visualization to optimize light setup in a scene, finding the causes of and resolving rendering artefacts, such as fireflies, as well as a path length contribution histogram to evaluate the efficiency of different Monte Carlo estimators.

Keywords: global illumination, rendering, visual analytics, visualization

ACM CCS: •Human-centered computing → Visual analytics; Visualization toolkits; •Computing methodologies → Ray tracing

1. Introduction

Physically based rendering techniques are the state-of-the-art technique for realistic light transport simulation. For lighting planners or rendering engineers, fast feedback is of high importance to observe and analyse changes to the scene; be it for aesthetic purposes, as in advertisement or movie production, lighting design as in automotive engineering to properly illuminate the environment, architecture, to increase comfort or follow legal requirements of work environments [Rel], or biology to optimize plant illumination for improving photosynthesis and growth. All these applications require physically-based light transport (PBLT) [Kaj86], the industry standard for synthesizing realistic images.

Unfortunately, for complex scenes, even the rendering of a single image can take hours of compute time. Sometimes, the images still contain rendering artefacts, such as fireflies caused by high-intensity samples. Causes can be either an underestimated directional probability, a too early path termination through Russian Roulette, or a presumably well working Monte Carlo (MC) estimator that does not perform as expected. Few tools exist which support the user to diminish, remove, speed up the rendering process or even analyse these artefacts. These tools are needed to gain more insight into the rendering process and its details.

Investigation of the rendering process is still in its infancy. Few tools exist and are often limited to very specific scenarios as it is often not clear what to look for. Researchers and engineers working in visual analytics often encounter the same problem when analysing a novel data set. Interactive data visualizations foster the knowledge gain. In analysing PBLT, we face a similar problem. Each light sample can be thought of as a data sample in a high-dimensional space. The dimensions include hit points, throughput, radiance, path length, etc. This provides a huge data set, even for a single image. In this work, we will address the interactive visualization of this data by combining a state-of-the-art renderer with classic interactive visual analytics tools.

Our approach is useful for rendering engineers who want to find out how light is distributed in a scene, how certain objects influence it, for finding out which parts cause significant noise and are not well handled by the renderer [KKG*14], or for those who want to improve rendering performance by steering the computation to focus on certain hard to render parts of the image. For other professionals, such as industrial designers or biologists, our proposed solution could help to design new products or optimize lighting situations by visualizing the light transport within a scene.

This work is an improved and extended version of [SAH*16]. We include all aspects and components of the previous paper for the sake of completeness and point out noteworthy differences where appropriate. In addition to the contributions, which are:

- a comparative light path visualization tool to improve the user's understanding of light transport within a single scene, including potential changes;
- selecting expressive samples from the set of available samples, which are too large to remain in memory or visualize sufficiently;
- interactive guidance of scenes, which are difficult to render,

we extend the previous version by:

- a more in-depth analysis with more complex scenes;
- a novel view to compare the effectiveness of different MC estimators based on histograms of the path length contribution of light samples.

Our visualization tool employs the following features:

- parallel coordinate plots (PCPs) [ID91] with established brushing interaction metaphors and subset selection;
- Two-dimensional (2D) and three-dimensional (3D) heat maps provide analysis capabilities within a spatial context;
- animated light path trajectories to illustrate light distribution;
- change visualizations to investigate the effect of scene variations;
- importance-sample editing capabilities to improve convergence of the rendering process;
- a histogram analysis tool to compare the effectiveness of different MC estimators, both globally and locally for a single image.

If not stated otherwise, we use a unidirectional path tracer in our experiments though the presented techniques are applicable to any path space sampling technique.

2. Related Work

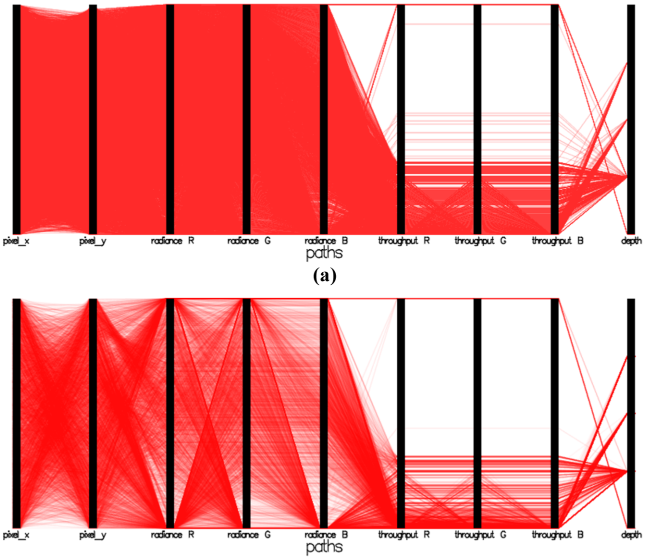
In PBLT, MC methods [CPC84] solve the rendering equation [Kaj86] numerically by drawing and integrating samples from an appropriate probability distribution function. Each sample constitutes a light path from the camera to a light source within the scene [Kaj86, VG97]. While physically correct, even state-of-the-art MC methods [KKG*14, HEV*16] and efficient ray-tracing engines [WFWB13] are far from real-time performance in complex scenes. Consequently, it is difficult for designers and rendering engineers to optimize their scene as any changes require a costly re-rendering.

Though being of interest for several years now, the visualization of light transport was usually focused on special-purpose visualizations. Signal-processing frameworks visualized the light frequency content and its change upon interaction with materials [DHS*05]. Visualization of light rays as geometric primitives helps to understand light propagation [Rus99]. The importance of filtering rays according to attributes or type was shown in [LP14] and made possible by recording the ray state [GFE*12]. On a higher level of abstraction, spherical plots and particle flow tools allow selective inspection of light transport [RKRD12]. Edge bundling [HvW09] applied to light path visualization reduces visual clutter and supports artists in path re-targeting [SNM*13]. Comparison of light transport in different scene setups has so far not been handled.

Several visualization techniques aim at exploring high-dimensional data [TM04]. One important technique within this field is dimensionality reduction. Within light transport, this has been used on quantities such as the irradiance vector field indicating the dominant light direction [CWW11] or finite-time path deflection [ZAD15]. while useful in the broader scope, such information compression may aggravate the analysis within complex scenes if only slight changes take place.

We make use of PCPs [ID91, HW13] together with interactive dimension reordering to visualize our attribute space. PCPs are well suited for our purpose as PCPs scale linearly with the number of dimensions. A PCP depicts K-dimensional data by displaying K axes in a parallel arrangement. A K-dimensional data point corresponds to a polyline connecting all axes (Figure 1). Visual clutter stemming from large data sets is a common problem with PCPs which are traditionally tackled with density-based techniques [MW91, HW09]. For up to a few thousand data points, techniques such as edge-bundling [PBO*14], hierarchical clustering [War94] and other visual clutter reduction techniques [ED07] prove useful but have not been tested for data sets such as ours consisting of several million data points. It is also unlikely that these techniques still provide interactive feedback with large data sets. We therefore propose to apply a sample-reduction technique beforehand. PCPs have been used before to visualize photon distributions and their attribute space [SJL15]. However, PCPs have not been used for comparative visualization.

In the field of comparative visualization, LiteVis [SOL*16] provides features to compare surface changes for interactive lighting design. In contrast to our approach, their technique builds upon virtual point lights [LTH*13] for fast feedback, which is a biased technique, and is restricted to surface measurements, whereas our



(a)

(b)

Figure 1: *Over-plotting problem with parallel coordinates. (a) A PCP with* 200 000 *data points. (b) A PCP with* 5000 *representative data points. In both figures, lines are drawn semi-transparently, but the overall distribution of the data is much more apparent in (b), whereas this distribution is almost invisible in (a) due to over-plotting.*

technique allows arbitrary scene changes and can be used to optimize scene arrangements and rendering. While inverse rendering [PP03] might be able to solve some of these problems, more fine-grained information visualization and manual inspection is often required.

3. Visual Analytics for Improved Physically Based Rendering

In the following, we will give an overview of our comparative, interactive visualization approach to visualize light transport. We collect, pre-process (Section 3.1) and visualize light transport data, including data that are usually only temporary and traditionally removed during the rendering process though it provides valuable information in our tool. The system provides interactive visual feedback and respective tools for interaction (Section 3.2). We show several useful applications in Section 4.

We first describe a typical use case: While rendering a scene, we collect data which are then visualized as a PCP. The user may then select subsets using brushing on the light path attributes. These selections may then guide further sampling for faster convergence. In our approach, no tedious selections in image space are required, which is the traditional approach and can be difficult or even impossible if samples outside the view have to be selected. While analysing standard render-time statistics can discover rendering problems, they are usually insufficient to resolve them. A good example are firefly artefacts (high-energy samples), which are easy to detect but removal of their cause is often difficult with traditional techniques, though simple with ours. We also provide comparative visualizations to visualize the changes in light transport before and after a scene edit. As our visualizations are updated on the fly, the user gets almost instant feedback without the need to wait for the renderer to finish. It should be mentioned that our current tool is only a proof-of-concept and merely shows the potential of a professional tool combining visualization and PBLT.

3.1. Data

During the rendering process, we collect several data for each light sample which is described below. This provides us with a high-dimensional data set that we use for exploration. We assume that the reader is familiar with the common terminology in PBRT; otherwise, we refer the reader to the textbook by Pharr and Humphreys [PH10].

For each light sample, which constitutes a *path* from a light source to the camera, we collect the following properties:

- *pixel position*,
- *exitant radiance*,
- *throughput* (computed from the bidirectional scattering distribution function (BSDFs) and pdfs along the path) and
- *depth* (number of intersections along the path).

Second, for each *intersection* of a ray with the scene, we collect:

- *position*,
- *exitant radiance*,
- *object identifier*,
- *bounce number* and
- *interaction type*, which can be either reflection or refraction.

Finally, for each *light source*, we collect its

- emitted *radiance* along each sample and the
- *light identifier*.

Note that additional attributes could be added easily. We display these three groups in separate PCPs (Figure 2).

3.1.1. *Data reduction*

Rendering even a single image with standard resolution requires millions of light samples which would clutter the visualization and limit interactivity. We therefore apply a data reduction technique to select and visualize only N samples. We empirically found a range of $5000 < N < 10\,000$ to be sufficient for our test cases, as it provides a good trade-off between memory consumption and resolution. These samples should faithfully represent the true underlying distribution of all samples. For this, we create histograms for each data dimension during rendering, counting the number of light samples for each bin and updating them whenever new samples are computed. For every $M = k \cdot N$ samples rendered, we create k random subsets of N samples each (in practice $k = 5$), compute their respective histograms H_N^j and compare these to the true distribution of all samples H_M to determine the closest set according to the metric:

$$D(H_N, H_M) := \sum_{i=1}^{I} \left| \frac{m_i}{M} - \frac{n_i}{N} \right|, \qquad (1)$$

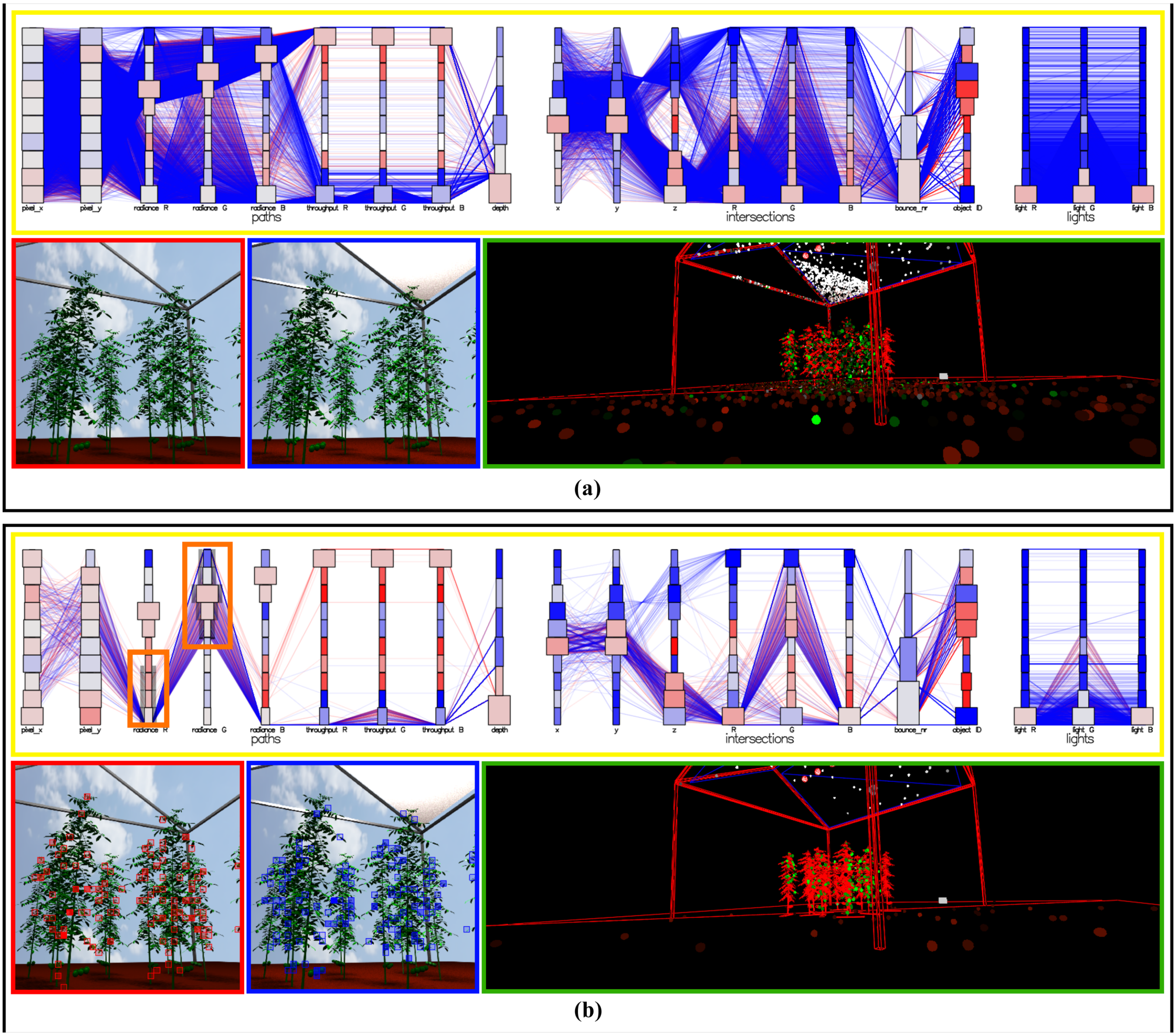


Figure 2: *Overview of our visualization tool. The parallel coordinates plot (yellow), the render views (red and blue like the colours in the parallel coordinates plot), and the scene view (green). The scene view represents meshes of the red and blue scenes with their respective colour, the point light sources being red spheres. (a) Visualization of the full data set. (b) Brushing of light paths that have high radiance values in the green component and low radiance values in the red component (indicated by the orange rectangles on the axis).*

where I is the number of discrete bins (in our case 10 per attribute), and m_i and n_i are the numbers of occurrences in bin i with respect to the true and reduced distribution. The computed distance is an estimate for the goodness of fit of each sample set, though we did not investigate the quality further as selecting the true best set is unfeasible. Figure 1 shows an example of our data reduction with a PCP. As selecting only subsets from a distribution of samples can never guarantee that all *important* samples are selected, we additionally allow constraining the sample selection to set regions of interest (Section 3.2).

3.2. Visual interface

Our visual interface consists of four components: the PCP for global data exploration (Section 3.2.1), the render view for image-space exploration (Section 3.2.2), the scene view for object-space exploration (Section 3.2.3) and the estimator comparison view (Section 3.2.4). We provide brushing-and-linking for all views. For comparing two scenes, our tool offers colour-coded differences or side-by-side views. Linking together the 2D and 3D views of the data merges scientific and information visualization in one powerful application.

Figure 2(a) shows an example of our tool, comparing two scenes representing a simple greenhouse (red and blue frame). They differ by reflective curtains on the upper windows of the greenhouse, placed in the right scene (blue frame). The PCP (yellow frame) provides a quick overview of the data. In Figure 2(b), we brushed the paths that contain high radiance values in the green channel and low radiance values in the red channel (orange rectangles). Both render views highlight the corresponding image parts. For example, these selections could be used to provide a probability distribution for adaptive sampling [PH10] over the image plane for noise reduction in hard to render image parts or for firefly elimination.

3.2.1. *Parallel coordinates plot*

For display of the selected light path data, we make use of PCPs [ID91] as they work well for up to 20 dimensions and provide simple ways to select subsets using brushing metaphors (see, e.g. [REB*15] for a comparison of different brushing techniques). Recall from Section 3.1 that the data of our sampled paths consist of multiple intersections, which may be associated with multiple light sources. Because of these N-ary relationships, we use three separate PCPs which are then linked together: one for paths, one for its constituent intersections and one for light sources.

To facilitate comparing scenes before and after an editing operation, we render the data for both scenes into the same PCPs using colour mapping (red and blue) to visually distinguish them (yellow frame in Figure 2).

We make use of binning [HLD02] in the PCPs to further illustrate the amount of data points within discrete ranges of each dimension. The width encodes the number of samples in each bin. The colour represents the ratio of samples between the two scenes. Let $H_1(b,x)$ denote the value in bin b for data dimension x in the first scene and $H_2(b,x)$ for the second scene. The colour of the bin is then based on the following computation:

$$H_{\text{diff}}(b,x) = \frac{(H_1(b,x) - H_2(b,x))}{(H_1(b,x) + H_2(b,x))}, \tag{2}$$

with $H_{\text{diff}}(b,x) \in [-1,1]$. This value is then mapped to a divergent colour map of red–grey–blue, seen in yellow frame in Figure 2. A saturated red bin indicates that all data points within this bin belong to the first scene, whereas blue stands for the second scene. A grey colour is used to indicate a bin whose ratio is balanced.

3.2.2. *Render view*

The render views (red and blue frames in Figure 2) display the current result of the rendering processes of two different scenes. These views are updated progressively as the rendering proceeds. The render view serves the purpose to show the rendered image itself but also to visualize the collected data in 2D image space with heat maps.

Figure 3 shows the Cornell box with four heat map visualizations. In Figure 3(a), the throughput distribution is displayed with the hot body colour map, which shows high values around light source and boxes. In Figure 3(b), the same colour map is used to visualize the number of intersections (depth) and in Figure 3(c) the radiance distribution. In addition, light paths that are brushed in the PCP can be visualized with a semi-transparent monochromatic heat map, as shown in Figure 3(d) where only light paths with strong radiance contributions in the green channel are selected. As only a subset of the original samples is saved, we use a coarse grid instead of the original pixels to depict the heat map. The opacity is normalized so that the grid entry with the most samples is fully opaque, while the transparency of the other selected areas is scaled linearly.

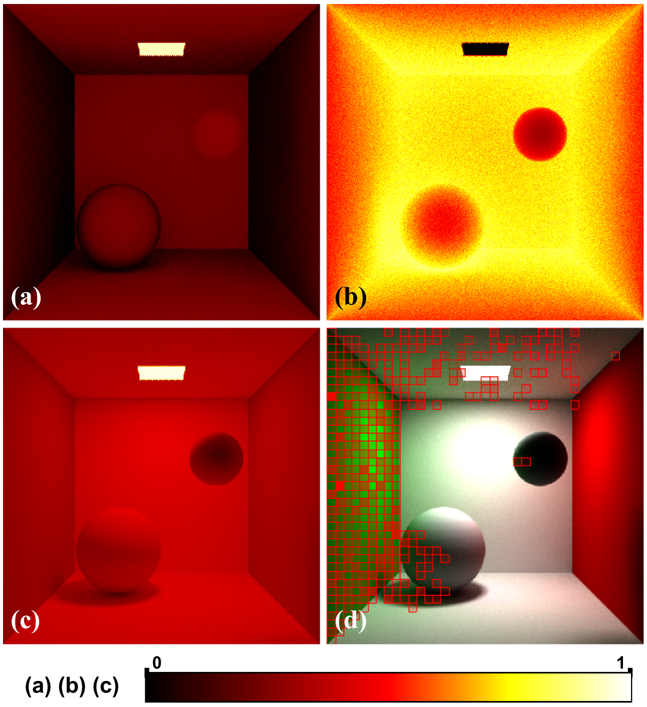


Figure 3: *Heat map visualizations of the (a) throughput, (b) number of intersections, (c) radiance and (d) brushed light paths. For each heat map, values are normalized between 0 and 1.*

Scribbling within the render view allows to select irregularly shaped image regions. Due to linking, the other visualizations are updated and only show sample data from these selected regions.

3.2.3. *Scene view*

The scene view (green frame in Figure 2) is for exploration and visualization of the data in 3D object space. As light samples can be considered as light rays or photons flowing through the scene, displaying their motion paths intuitively conveys the light transport within the scene. Intersection points are coloured according to the received energy, whereas paths are coloured according to the gross accumulated energy. If the user is interested in the difference at each bounce, then they may select to colour the light paths according to the number of bounces from the light source. We render animated line segments to visualize the light paths, as polylines connecting the intersection points would cause clutter and a loss of orientation.

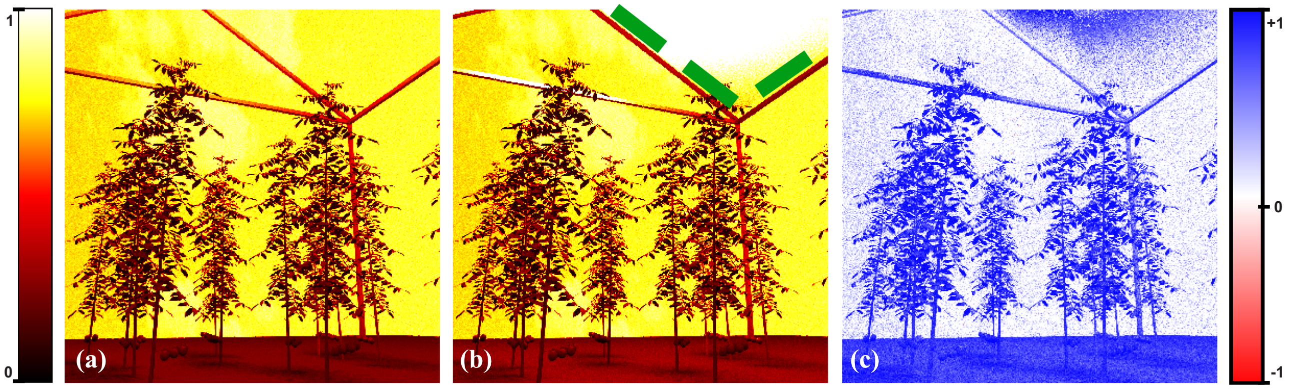


Figure 4: *3D heat map visualization of the energy distribution in a scene (a) without and (b) with reflective curtains (hinted in dashed green lines for better readability) that reflect artificial light inside the greenhouse. (c) Visualization of the difference of the energy distributions of both scenes with a divergent red–blue colour map.*

Heat Map We create a 3D heat map of the energy distribution using a sparse voxel octree [LK11]. Visualizing the energy on surfaces can be helpful for scene design and object placement. While 3D heat maps are not a new contribution, their usage in this context is, to the best of our knowledge. To build the octree, we start with a root node voxel containing all intersection points. This is recursively partitioned into eight child voxels until a stopping criterion is reached (number of intersections or maximum depth). The average of the intersection energy over all (the selected) colour channels within a voxel is mapped to the voxel colour using a hot-body colour map. Alternatively, for comparing two scenes, the difference of the energies is mapped to a red–grey–blue divergent colour map.

Figure 4 depicts an example of a 3D heat map for the comparison of the two scenes in Figure 2. Figure 4(a) shows the energy distribution in the scene where the upper windows are left unobstructed, whereas Figure 4(b) shows the result with reflective curtains that scatter back the artificial light produced inside a greenhouse. Figure 4(c) shows the difference of both setups using the divergent colour map. In the latter, blue/red colour indicates that the scene with/without the curtains receives more energy. The plants receive more energy in the presence of the curtains, as indicated by the dominant blue shade.

Interaction The scene view additionally provides a convenient way to select subsets of light paths that interact with certain objects. As each object in the scene has a unique identifier the user can select light paths that only interact with certain objects by clicking on them. We offer three different selection mechanisms. First, in the *path-selection mode*, the user selects objects consecutively and all light paths interacting with these objects in the selected order are chosen. For example, selecting only paths hitting an object A and then object B. Second, the *shadow selection mode* allows the user to select light paths that intersect with the scene at positions that lie in the shadow of an object according to a light source. Third, a region of interest (ROI) selection, for which a gizmo is placed in the scene to collect localized information [RKRD12]. We extend the use of the gizmo for our comparative visualization tool. By placing two gizmos, one in each scene, we can restrict the samples to those intersecting with the gizmos and make a comparison between these two distinct ROIs. Figure 5(a) shows an example where all paths are visualized for the Cornell box of Figure 3, whereas in Figure 5(b), only those paths that intersect the gizmo are displayed.

3.2.4. *Estimator evaluation view*

To show the extensibility of our tool, we introduce a novel view called estimator evaluation view, Figures 9 and 10, which we use to compare the effectiveness of different MC estimators and their path generation process. To date, the most common technique to compare estimators is usually by looking at the results of the estimators after equal time or equal number of samples per pixel. While metrics such as the mean squared error (MSE), the relative mean squared error (relMSE) or the structural similarity index (SSIM) give valuable information about the variance of an estimator, they give little insights into why one estimator performs better than another, or why one is more efficient. The efficiency of an MC-based path-tracing estimator is defined by its capability to sample/generate paths with a probability proportional to their contribution to the final estimate and by the time needed to generate one of these paths. If we look at this from the perspective of the path space formulation [VG97], which describes all possible paths in a scene up to a maximum path length of L, an ideal path generation process would sample paths, where the number of contributions at a path length n would be proportional to the actual radiance contribution at this path length. To visualize the ability of an estimator to generate paths with this optimal contribution characteristic, we measure the relationship between the contribution histogram H_c and the positive contribution count histogram H_{pc} for each pixel i by using their summed squared difference:

$$M(i) = \sum_{l=1}^{L} \left(\frac{H_c(i,l)}{\bar{H}_c(i)} - \frac{H_{pc}(i,l)}{\bar{H}_{pc}(i)} \right)^2 . \tag{3}$$

The contribution histogram H_c contains the radiance contribution for each pixel i at a given path length n to the final radiance value of the pixel, while the positive contribution count histogram H_{pc} contains the number of radiance contributions at the path length

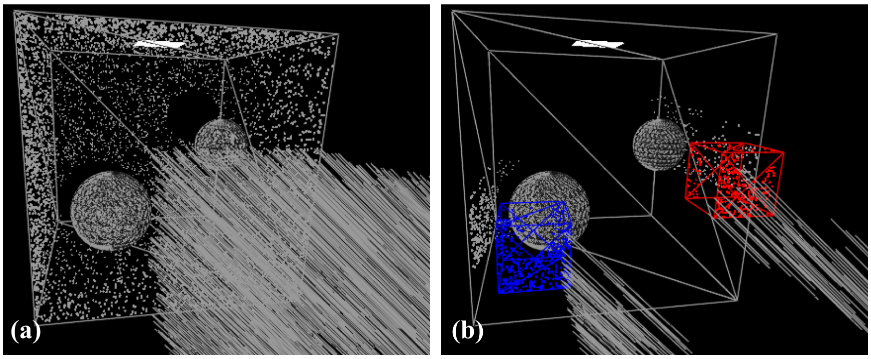


Figure 5: *ROI-based path selection with gizmos. (a) Visualization of all light paths. (b) Visualization of the light paths that intersect the gizmos (red and blue cubes).*

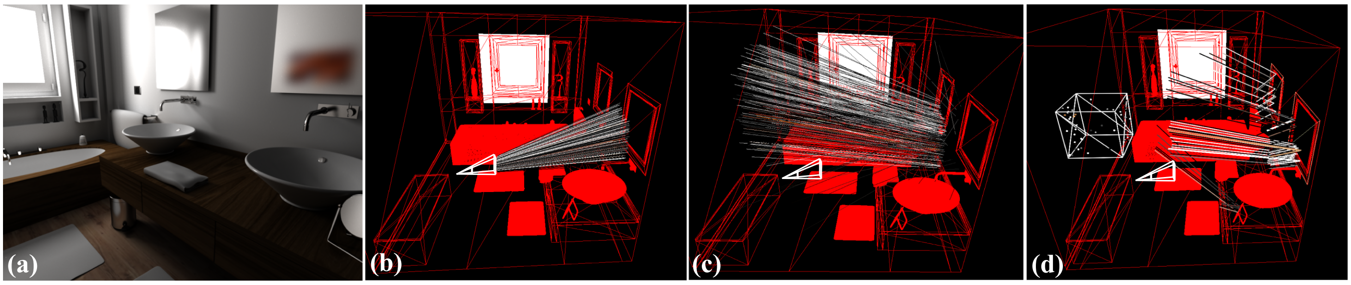


Figure 6: *We place a painting in the bathroom that is only visible in one of the mirrors (a). We select and follow paths that intersect the mirror (b) and (c) to find a potential position for the image. In (d), we verify this placement, by creating a gizmo at the estimated position and selecting light paths that pass through this gizmo.*

n. To make the metric independent from the intensity of the estimate of each pixel or the number of samples used, we normalize all individual histogram bins using $\bar{H}_{\mathrm{c}}(i) = \sum_{l=1}^{L} H_{\mathrm{c}}(i,l)$ and $\bar{H}_{\mathrm{pc}}(i) = \sum_{l=1}^{L} H_{\mathrm{pc}}(i,l)$. The contribution histogram is generated through tracking the contribution for a maximum path length of L during the rendering process of the ground truth result. It is the same for each estimator, while the positive contribution histogram is tracked during the execution of each individual estimator.

A summarized view of the efficiency of the estimator can be generated by comparing the average contribution histogram over all pixels to the averaged contribution count histogram over all pixels (Figure 10). In Section 4, we will show how these two new visualizations can give additional insights into the behaviour of different MC estimators based on path tracing.

4. Results and Discussion

We have combined the previously proposed visualization techniques into a custom-built C++ OpenGL application and adapted the unidirectional path-tracing EMBREE renderer [WFWB13] for our purposes. For the estimator evaluation, we used data which we precomputed with the PBRT renderer [PH10] and dumped to disk, as the technique requires a reference image. In this section, we will discuss various example applications and the obtained results.

4.1. Scene optimization

First, we give an example of how our tool can aid in understanding the light distribution within a scene. In this example, we want to place a painting in the bathroom scene shown in Figure 6(a). However, this painting should not be directly visible, but only visible in one of the mirrors. A simple trial and error placement and preview rendering is ineffective, because convergence in highly reflective areas can be slow, prohibiting interactive feedback.

Therefore, we employ our proposed visualization techniques to select and follow paths that intersect the mirrors from the camera (Figures 6 b and c). This gives us a general idea on where to place the painting. To verify this assumption, we place a gizmo at the estimated position (Figure 6 d). We can now select all paths that pass through the gizmo and can thus verify that they connect to the camera through the mirrors.

The same interaction pattern could also be used for refraction by selecting and visualizing the light flow through the selected pixels. In our previous work [SAH*16], we showed that this is also useful in other areas, such as lighting optimization.

4.2. Rendering optimization

In the following, we show how our tool can be used to directly influence the rendering process to increase the convergence rate, and thereby speeding up rendering times.

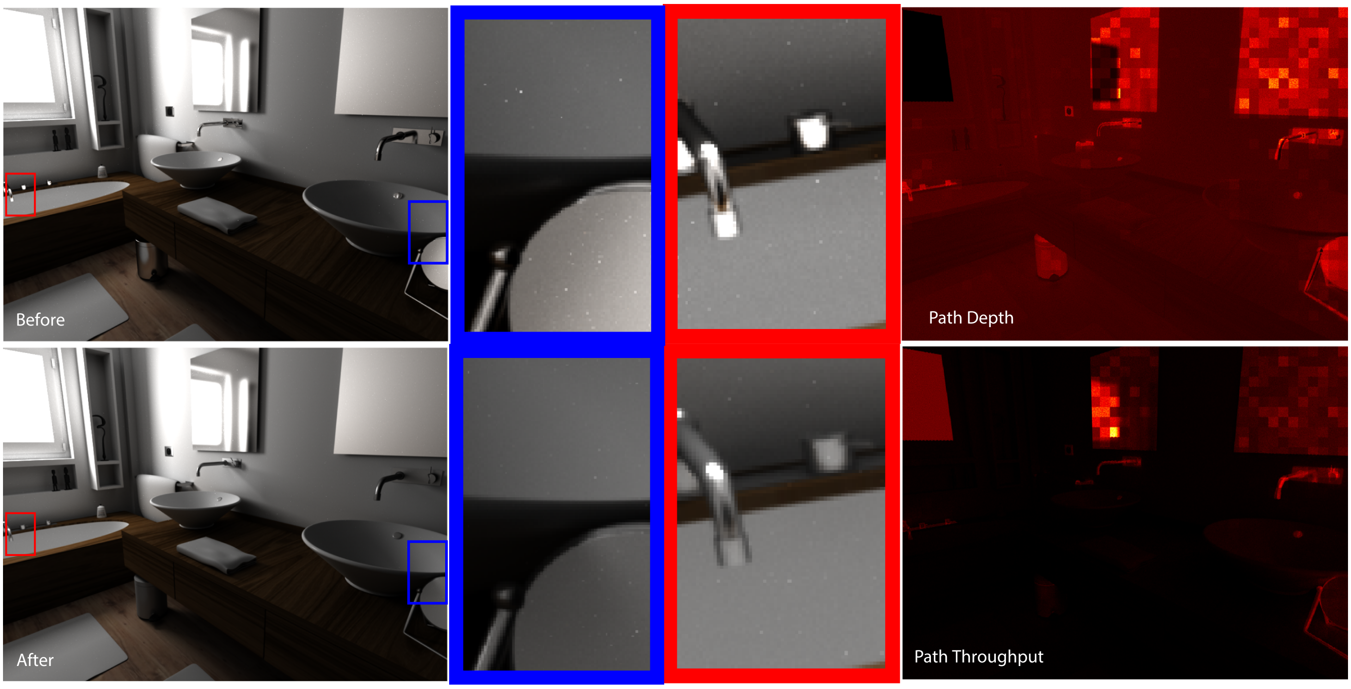


Figure 7: *We find causes of fireflies in the bathroom scene with the path depth and throughput heat map visualizations. We then make an informed choice on how to reduce the fireflies: We reduce the specularity of the faucets and the wastebin, but do not change the look of the mirrors. This way we manage to reduce the amount of fireflies, but do not significantly alter the resulting image.*

Firefly Detection Fireflies im MC renderings are usually caused by high-energy samples with a small probability resulting in very bright pixels which are difficult to compensate for with additional samples. Finding the cause for these fireflies can be difficult as the cause may lie outside the rendered view. Techniques like clamping or path skipping result in physically inaccurate renderings which may or may not be acceptable. A more useful choice would be to inform the user of the causes and let him/her decide how to handle them, e.g. by changing the materials, geometry or camera parameters.

Figure 7 shows our bathroom scene that contains several fireflies. With our heat map visualization of the path depth and throughput, we can easily identify the cause of fireflies in this scene: the two mirrors, the faucets and the wastebin. Note that we could have achieved similar results by brushing high-energy paths in the PCP and that we can also detect causes of fireflies outside the rendered view. With the cause of the fireflies known, the user is now able to make an informed choice on how to change the scene. In our example, we do not want to change the mirrors since they are an important part of the image. On the other hand, the faucets and the wastebin do not contribute much to the look of the image, so we reduce their specularity as shown in the bottom row of Figure 7. Note that we have created equal time images for the before and after results. Although the improvements become obvious in the close-ups, we did not remove all fireflies. At the same time, our changes were minimal and did not significantly change the end result. An iterative refinement could then be applied to remove further firefly causes.

It should be noted that due to the sub-sampling, not all fireflies will be detected. As we are mostly concerned with the origin of the fireflies and not the fireflies themselves, detecting all fireflies is generally not necessary.

User-Guided Rendering Convergence of an image is highly affected by the distribution of samples as it is preferable to create more samples for more difficult to render areas. We show how to use our tool to assign more samples to those areas containing more complex path interactions to improve the performance and quality of the output.

The bathroom scene in Figure 8 contains a lot of variance due to many specular objects. Previously, we have already identified the mirrors as a cause of fireflies. But the mirrors are also responsible for the generally slow convergence, as shown in the inlets in Figure 8. To improve convergence, we select paths that intersect the mirrors. These one-click selections are then automatically transformed into a probability distribution function over the image plane to focus further samples on the selected areas (Figure 8 a).

The guided sampling reduces the MSE in our scene compared to uniform sampling and additionally improves the structural similarity measure (SSIM) [WBSS04] compared to the reference shown in Figure 8. This is especially noticeable in the inlets, where the guided sampling is close to the reference. Focusing samples on difficult to render elements of the image reduces the sample count in other areas and therefore increases the error in these areas, if the same sample budget is used. However, the reduced MSE and perceived quality show that our technique of user-guided rendering is favourable in these complex rendering situations.

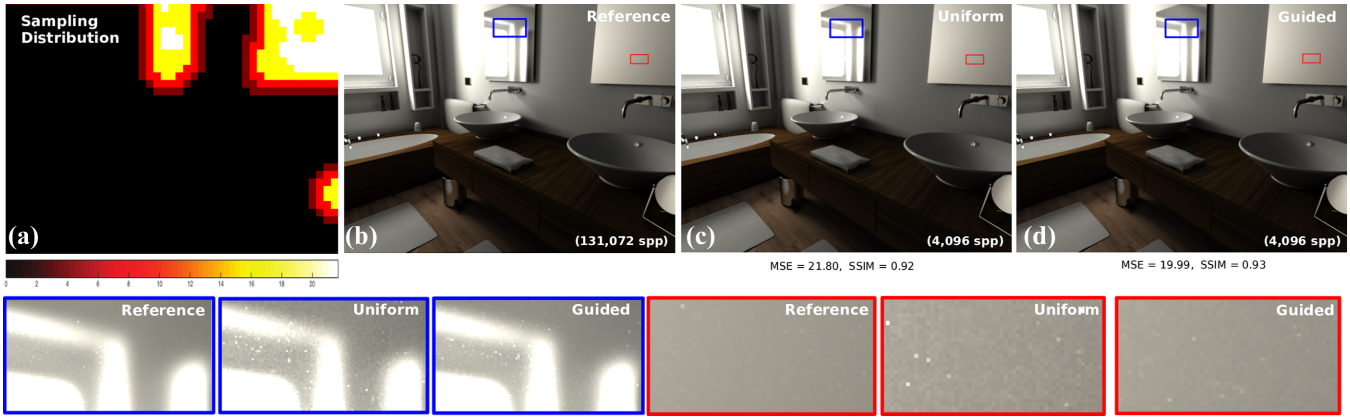


Figure 8: *An example application of our user-guided rendering technique. (a) Visualization of the sampling distribution used to guide the rendering based on light paths that intersect the mirror. (b) Reference image. (c) Rendering with uniform sampling. (d) Our user-guided rendering technique. In the bottom row, we show insets of the scene for better readability.*

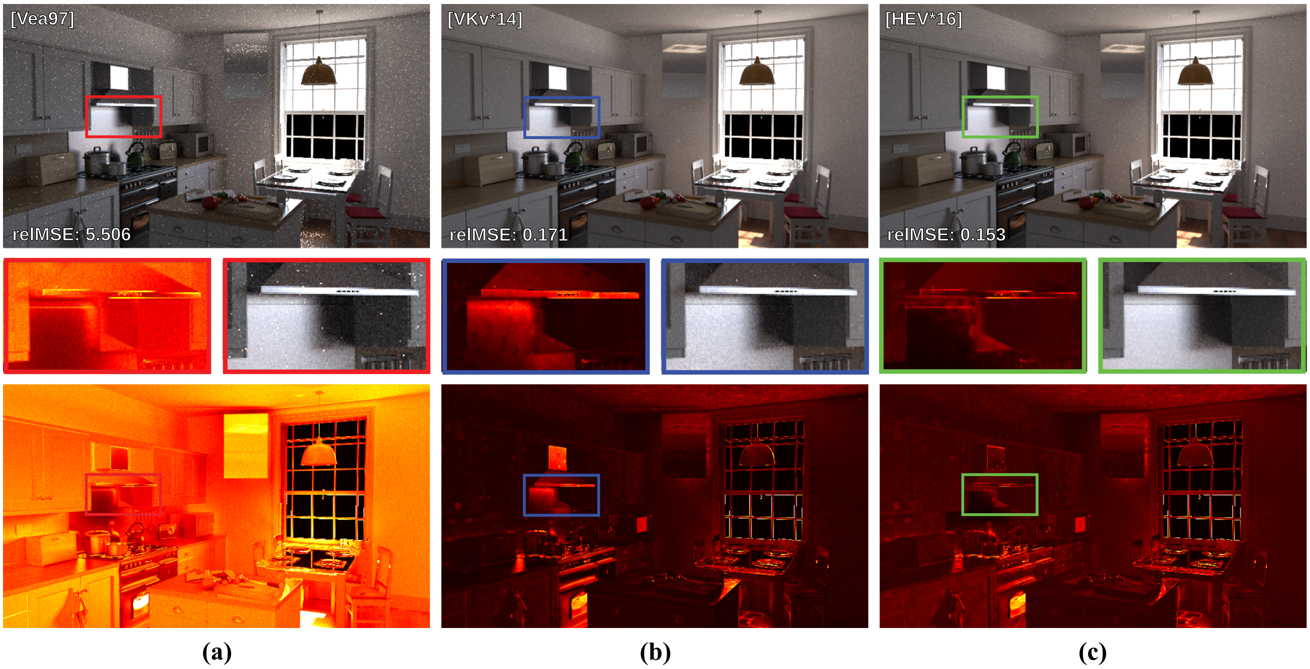


Figure 9: *Per-pixel estimator evaluation: Comparison of the relationship between the positive contribution count per path length for each pixel and the actual contribution at each path depth. (a) Standard MIS path tracer [Vea97], (b) illumination-guided path tracer [VKv*14] and (c) product-guided path tracer [HEV*16]. For a standard path tracer (a), it is difficult to distribute the amount of contributing paths according to the radiance distribution of the scene. The two guiding-based methods (b–c) have additional information about the radiance distribution and direct the paths into areas of high importance. Using incoming illumination only for guiding can lead to inefficient decisions at glossy surfaces (b). Through the integration of the product with the BSDF, these shortcomings can be overcome (c). The same colour map as in Figure 8 is used.*

In our previous work [SAH*16], we showed how we can also use this technique so steer the sampling towards caustics, which are difficult to render for a classic path tracer.

Estimator Evaluation For comparing different MC-based estimators, we use our novel estimator comparison view (Section 3.2.4) to gain additional insights beside the relative MSE of their estimates, about each estimators path generation behaviour. Although we used our tool in the previous examples to improve or change the current scene (or rendering), the estimator comparison view instead shows a more abstract use of our tool to investigate the effectiveness of rendering algorithms or estimators, which requires a pre-rendered

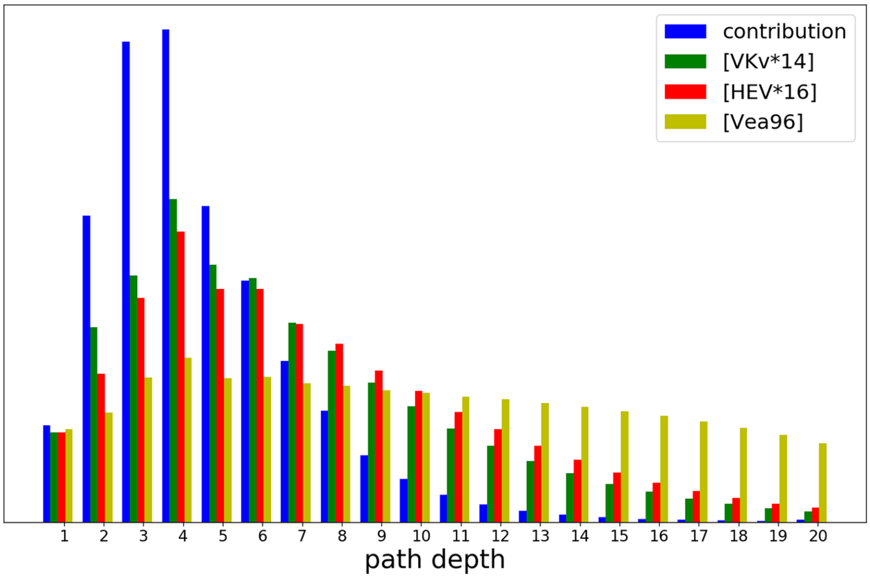


Figure 10: *Summarized estimator evaluation: Comparison of the histograms for the relative contribution at each path depth of the scene (blue) and the positive contribution counts of different estimators from Figure 9. The histograms are averaged over all pixels. While the histogram of standard path is more or less uniform, both histograms of the guiding-based methods are closer to the optimal contribution distribution.*

scene. To demonstrate the potential of the method, we compare two different state-of-the-art path-tracing algorithms, which are based on path guiding, to a standard multiple importance-based path tracer [Vea97]. During the path generation process, both guided methods use multiple importance sampling (MIS) and the balance heuristic to sample a new direction from either the BSDF or a guiding distribution. The first, introduced by Vorba *et al.* [VKv*14], uses an approximation of the incoming illumination as guidance, while the second, from Herholz *et al.* [HEV*16], uses an approximation of the product between the BSDF and the incoming illumination.

Figure 9 shows the per pixel visualization of our estimator evaluation, while Figure 10 shows a summary of the sampling behaviour of the estimators by averaging the path statistics over all pixels and present it in one histogram. For our test, we use the same modified ‘Country Kitchen’ scene as used by Müller *et al.* [MGN17]. The scene is rendered using 1024 samples per pixel and a maximal path depth of 20. For path termination, the fixed weighted window Russian Roulette (RR), as described by Vorba *et al.* [VKv*14], is used. This type of RR prevents early path termination based on the current throughput of the path and therefore paths are most likely only terminated when they leave the scene and reach the environment map or when they reach the maximum path length. Both guiding-based methods use a BSDF sampling weight of $\alpha = 0.25$.

From Figure 9(a), we can see that a standard path tracer is not able to efficiently sample proportional to the complex light transport in the scene. This can be seen, e.g. in the caustic on the floor and its reflection in the mirror. Since new directions are only sampled via the BSDF, the generated paths are unlikely to directly reach the environment light through the window and non-contributing light bounces are computed. This leads to an almost uniform distribution of the positive contribution count histogram across all path depths (Figure 10, yellow). The illumination guided path tracer [VKv*14] (Figure 9 b) guides sampling based on the incoming illumination and therefore captures the caustics better. It still has problems on glossy surfaces, where the importance of the incoming illumination diverges from the importance of the actual reflected illumination. The product-guided path tracer [HEV*16] (Figure 9 c) uses the product of the BSDF and the incoming illumination approximation to overcome this shortcoming of the illumination-based guiding. This is visible on the rough surfaces behind the stove or on the top of the cupboards. In the overall histogram in Figure 10, both estimators generate a similar positive path count distribution, which more closely resembles the actual radiance distribution then the standard path tracer does. Combining our global and local estimator views can clearly guide the analyst to the strength and weaknesses of each of the estimators. We predict this tool to be useful when determining the efficiency of new estimators and to gain further insights into the rendering process.

5. Conclusion and Future Work

In this work, we investigated the usage of visual analytics tools to analyse and improve rendering in PBLT. In addition to classic rendering, we have shown real-world examples from areas as diverse as engineering and biology/agriculture that benefit from our developed techniques. We have shown the versatility of incorporating parallel coordinates plots and gizmos to select and investigate specific light paths. An effective data reduction technique allows for interactive feedback. 2D and 3D heat map visualizations assist in further investigation to detect and correct critical constellations and to guide the sampling for more rendering efficiency. We have shown the extensibility of our approach, including the possibility to analyse the quality of MC estimators, and we believe that there are many other potential applications and extensions to further investigate, e.g. at the moment, our technique is limited to static scenes without participating media. A useful extension would also be to investigate the sampling density throughout the scene to guide the sampling not only in image space but also within the 3D scene. This investigation could help in other areas such as optimizing lighting in office spaces. We would also like to test whether common dimensionality reduction techniques such as t-distributed stochastic neighbor embedding (t-SNE) or principal component analysis (PCA), which are application agnostic, could give further useful insights into the rendering process.

To further pursue the development of new extensions, we intend to make the code publicly available. More customization options and other visualization techniques, which illustrate information at more abstract levels, would further improve the applicability of our tool.

Acknowledgements

This work is part of the research programme 'LED it be 50%' (project number P13-20), which is funded by the Netherlands Organisation for Scientific Research (NWO) and supported by LTO Glaskracht, Philips, Nunhems, WUR Greenhouse Horticulture.

References

[CPC84] Cook R. L., Porter T., Carpenter L.: Distributed ray tracing. *SIGGRAPH Computer Graphics 18*, 3 (1984), 137–145.

[CWW11] Chajdas M. G., Weis A., Westermann R.: Assisted environment map probe placement. In *Proceedings of SIGGRAD* (Linköping, Sweden, 2011), Linköping University Electronic Press, pp. 17–25.

[DHS*05] Durand F., Holzschuch N., Soler C., Chan E., Sillion F. X.: A frequency analysis of light transport. *ACM Transactions on Graphics 24*, 3 (2005), 1115–1126.

[ED07] Ellis G., Dix A.: A taxonomy of clutter reduction for information visualisation. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1216–1223.

[GFE*12] Gribble C., Fisher J., Eby D., Quigley E., Ludwig G.: Ray tracing visualization toolkit. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2012), ACM, pp. 71–78.

[HEV*16] Herholz S., Elek O., Vorba J., Lensch H., Křivánek J.: Product importance sampling for light transport path guiding. *Computer Graphics Forum 35*, 4 (2016), 67–77.

[HLD02] Hauser H., Ledermann F., Doleisch H.: Angular brushing of extended parallel coordinates. In *Proceedings of IEEE Symposium on Information Visualization (INFOVIS)* (2002), IEEE Computer Society, pp. 127–130.

[HvW09] Holten D., van Wijk J. J.: Force-directed edge bundling for graph visualization. *Computer Graphics Forum 28*, 3 (2009), 983–990.

[HW09] Heinrich J., Weiskopf D.: Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1531–1538.

[HW13] Heinrich J., Weiskopf D.: State of the art of parallel coordinates. In *Eurographics 2013 - State of the Art Reports* (2013), M. Sbert and L. Szirmay-Kalos (Eds.), The Eurographics Association, pp. 95–116.

[ID91] Inselberg A., Dimsdale B.: Parallel coordinates. In *Human-Machine Interactive Systems*. A. Klinger (Ed.). Springer, Boston, MA (1991), pp. 199–233.

[Kaj86] Kajiya J. T.: The rendering equation. *ACM SIGGRAPH Computer Graphics 20*, 4 (1986), 143–150.

[KKG*14] Křivánek J., Keller A., Georgiev I., Kaplanyan A. S., Fajardo M., Meyer M., Nahmias J.-D., Karlík O., Cañada J.: Recent advances in light transport simulation: Some theory and a lot of practice. In *ACM SIGGRAPH 2014 Courses* (2014), pp. 17:1–17:6.

[LK11] Laine S., Karras T.: Efficient sparse voxel octrees. *IEEE Transactions on Visualization and Computer Graphics 17*, 8 (2011), 1048–1059.

[LP14] Lesev H., Penev A.: A framework for visual dynamic analysis of ray tracing algorithms. *Cybernetics and Information Technologies 14*, 2 (2014), 38–49.

[LTH*13] Luksch C., Tobler R. F., Habel R., Schw"arzler M., Wimmer M.: Fast light-map computation with virtual polygon lights. In *Proceedings of ACM Symposium on Interactive 3D Graphics and Games 2013* (2013), ACM, pp. 87–94.

[MGN17] Müller T., Gross M., Novák J.: Practical path guiding for efficient light-transport simulation. *Computer Graphics Forum 36*, 4 (June2017), 91–100.

[MW91] Miller J. J., Wegman E. J.: Construction of line densities for parallel coordinate plots. In *Computing and Graphics in Statistics*. A. Buja and P. A. Tukey (Eds.). Springer-Verlag, New York (1991), pp. 107–123.

[PBO*14] Palmas G., Bachynskyi M., Oulasvirta A., Seidel H. P., Weinkauf T.: An edge-bundling layout for interactive parallel coordinates. In *Proceedings of the 2014 IEEE Pacific Visualization Symposium* (Yokohama, Japan, 2014), IEEE Computer Society, pp. 57–64.

[PH10] Pharr M., Humphreys G.: *Physically Based Rendering, Second Edition: From Theory to Implementation* (2nd edition). Morgan Kaufmann Publishers Inc., Burlington, MA, 2010.

[PP03] Patow G., Pueyo X.: A survey of inverse rendering problems. *Computer Graphics Forum 22*, 4 (2003), 663–687.

[REB*15] Raidou R., Eisemann M., Breeuwer M., Eisemann E., Vilanova A.: Orientation-enhanced parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (2015), 589–598.

[Rel] Relux Informatik AG: Reluxsuite. http://www.relux.info/. Accessed 27 November 2015.

[RKRD12] Reiner T., Kaplanyan A., Reinhard M., Dachsbacher C.: Selective inspection and interactive visualization of light transport in virtual scenes. *Computer Graphics Forum 31*, 2 (2012), 711–718.

[Rus99] Russel J. A.: *An Interactive Web-Based Ray Tracing Visualization tool*. Master's thesis, University of Washington, 1999.

[SAH*16] Simons G., Ament M., Herholz S., Dachsbacher c., Eisemann M., Eisemann E.: An interactive information visualization approach to physically-based rendering. In *Proceedings of the Vision, Modeling, and Visualization Workshop* (Bayreuth, Germany, 2016), pp. 1–8.

[SJL15] Spencer B., Jones M. W., Lim I. S.: A visualization tool used to develop new photon mapping techniques. *Computer Graphics Forum 34*, 1 (2015), 127–140.

[SNM*13] Schmidt T.-W., Novak J., Meng J., Kaplanyan A. S., Reiner T., Nowrouzezahrai D., Dachsbacher C.: Path-space manipulation of physically-based light transport. *ACM Transactions On Graphics 32*, 4 (2013), 129.

[SOL*16] Sorger J., Ortner T., Luksch C., Schwärzler M., Gröller E., Piringer H.: Litevis: Integrated visualization for simulation-based decision support in lighting design. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (2016), 290–299.

[TM04] Tory, M., Möller, T.: Rethinking visualization: A high-level taxonomy. In *Proceedings of the IEEE Symposium on Information Visualization* (Austin, Texas, 2004), pp. 151–158.

[Vea97] Veach E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.

[VG97] Veach E., Guibas L. J.: Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 65–76.

[VKv*14] Vorba J., Karlík O., Šik M., Ritschel T., Křivánek J.: On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics 33*, 4 (2014), 101:1–101:11.

[War94] Ward M. O.: Xmdvtool: Integrating multiple methods for visualizing multivariate data. In *Proceedings of the Conference on Visualization'94* (Los Alamitos, CA, USA, 1994), IEEE Computer Society Press, pp. 326–333.

[WBSS04] Wang Z., Bovik A., Sheikh H., Simoncelli E.: Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (2004), 600–612.

[WFWB13] Woop S., Feng L., Wald I., Benthin C.: Embree ray tracing kernels for CPUs and the Xeon Phi architecture. In *ACM SIGGRAPH 2013 Talks* (2013), pp. 44:1–44:1.

[ZAD15] Zirr T., Ament M., Dachsbacher C.: Visualization of coherent structures of light transport. *Computer Graphics Forum 34*, 3 (2015), 491–500.